Yocto Layer CI Build and Test with GitHub Actions

Alex J Lennon, Dynamic Devices Ltd

Yocto Project Summit, 2022.05

A Little About Me

dynamic DEVICES LTD

- **Based in DoES Liverpool Makerspace**
- **Founded Dynamic Devices in 2004**
  - providing Embedded Linux and RTOS integration services
  - part of a “soup to nuts” eco-system to deliver concept, prototyping, manufacture, mechanicals, certifications, drop shipping, etc.
  - a Yocto Project Participant since 2014

- **Been working with “Embedded Linux” since 20th Century**

Maintaining meta-mono

…how it began

Maintaining meta-mono, how it began…

- Props to Autif Khan for creating the layer in 2012
- I got involved back in April 2012 as I was playing with Mono for Embedded Linux to port .NET Compact Framework apps from Windows CE
- I use Yocto extensively in my work and I wanted to do something to contribute back to the community
- Took the lead on maintainership in 2014

The Challenges

- In the early days new releases of Mono would often break in unexpected ways when cross-compiling
- Mono itself can take quite a while to compile
- I was “learning on the job” how to do this
- I found it quite slow going to follow a diff and email patch contribution mechanism when we hosted on https://git.yoctoproject.org/meta-mono
- It felt hard to build engagement from contributors

Some Attempted Solutions

- With the support of the Michael Halstead @YoctoProject I migrated to GitHub in 2019 to leverage PRs and Issues and so forth. (Mirrored back to git.yoctoproject.org)
- I started trying to understand how to add in simple tests to check that my builds were somewhat functional. For example sometimes the framework would build but Mono would fail to run apps

The Yocto Project Test Environment

- meta-mono/lib/oeqa/runtime/cases/mono.py

```
from oeqa.runtime.case import OERuntimeTestCase
…
class MonoCompileTest(OERuntimeTestCase):
…
    @OETestDepends(['ssh.SSHTest.test_ssh'])
    def test_executable_compile_and_run_cmdline(self):
        status, output = self.target.run('mcs /tmp/helloworld.cs -out:/tmp/helloworld.exe')
        msg = 'mcs compile failed, output: %s' % output
        self.assertEqual(status, 0, msg=msg)
        status, output = self.target.run('mono /tmp/helloworld.exe')
        msg = 'running compiled file failed, output: %s' % output
        self.assertEqual(status, 0, msg=msg)
        self.assertEqual(output, 'HelloWorld', msg=msg)
```

Maintaining meta-mono

Live… looking at Yocto ‘testimage’ tests

Reference Links

- The Yocto Project Test Environment Manual
  https://docs.yoctoproject.org/test-manual/intro.html
- Using the Quick EMUlator (QEMU)
  https://docs.yoctoproject.org/dev-manual/qemu.html
- meta-mono test cases
  https://github.com/DynamicDevices/meta-mono/blob/master/lib/oeqa/runtime/cases/mono.py

Trying to build in the Cloud

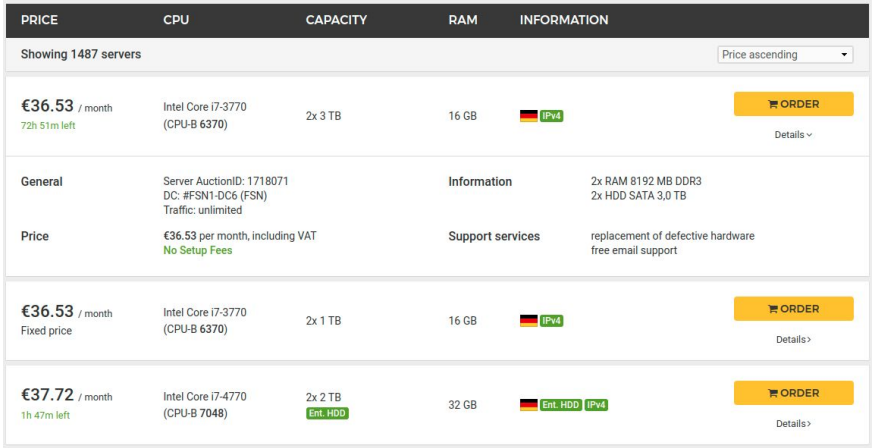
First Steps at Cloud Building

- I tried a number of hosted cloud service including AppVeyor, Travis CI. both **really helpful** to OpenSource
- **BUT** Yocto takes some grunt to build and build VMs are not super high powered and limit build time.

  My Local Build Box 8 cores, 32 GB, no limit
  Appveyor on Hyper-V 2 cores, 6 GB, 60 minute limit
  Travis CI 2 cores, 7.5GB, 50 minute limit
- Both **tried to help** but this costs money and they couldn't go much above 2 hours, when I needed 8-10+ (!!!)

Hosting my own servers with Hetzner and ProxMox

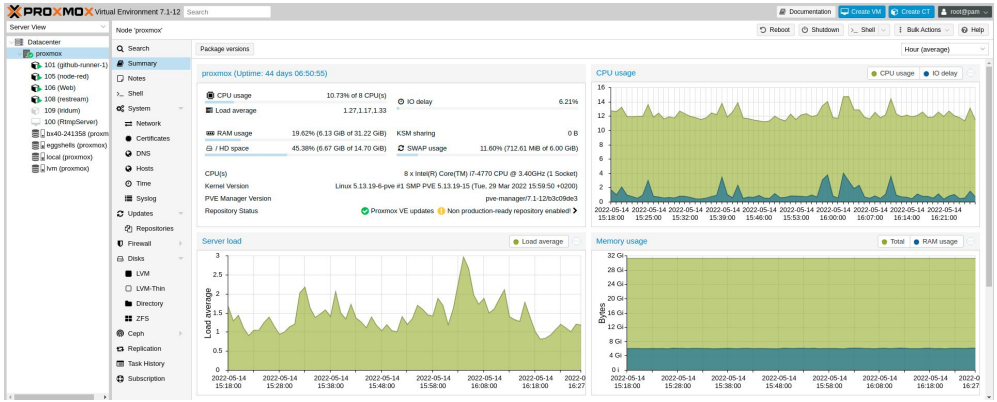
- The folks at DoES Liverpool recommended **Hetzner**
- Their **Server Auctions** are great, pre-loved hardware, fantastic service, great price



PRICE	CPU	CAPACITY	RAM	INFORMATION
Showing 1487 servers				Price ascending
€36.53 / month 72h 51m left	Intel Core i7-3770 (CPU-B 6370)	2x 3 TB	16 GB	IPv4 — ORDER — Details
General	Server AuctionID: 1718071 DC: #FSN1-DC6 (FSN) Traffic: unlimited	Information	2x RAM 8192 MB DDR3 2x HDD SATA 3,0 TB	
Price	€36.53 per month, including VAT No Setup Fees	Support services	replacement of defective hardware free email support	
€36.53 / month Fixed price	Intel Core i7-3770 (CPU-B 6370)	2x 1 TB	16 GB	IPv4 — ORDER — Details
€37.72 / month 1h 47m left	Intel Core i7-4770 (CPU-B 7048)	2x 2 TB Ent. HDD	32 GB	Ent. HDD IPv4 — ORDER — Details

Hosting my own servers with Hetzner and ProxMox

- They also recommended ProxMox
- Fantasic OpenSource Server Virtualisation Platform supporting VMs and Containers



ProxMox on Hetzner Servers

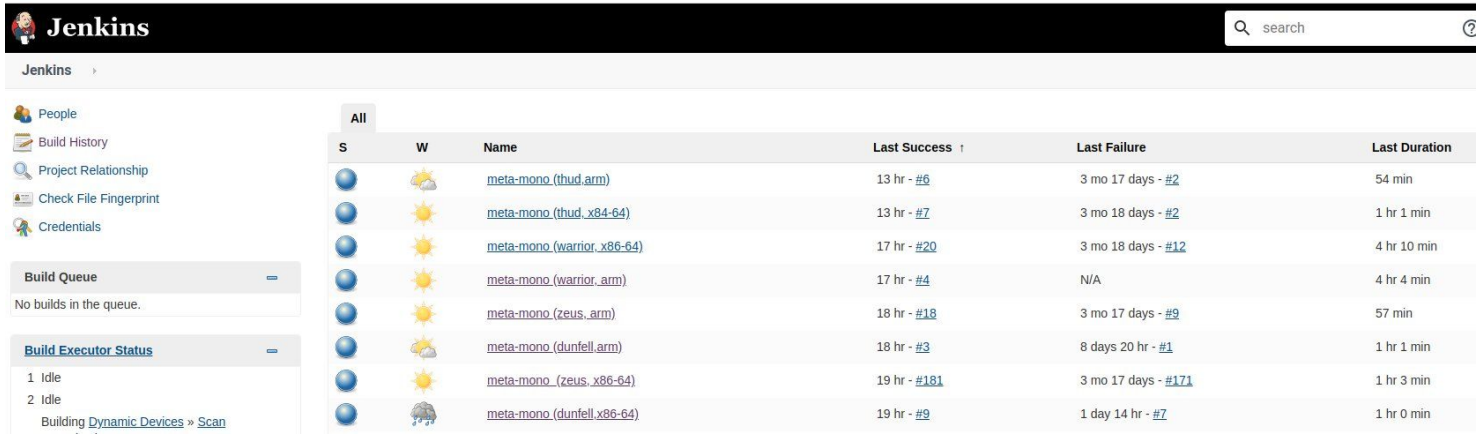
Live… looking at one of my ProxMox boxes

Reference Links

- Hetzner Server Auction
  https://www.hetzner.com/sb
- ProxMox
  https://www.proxmox.com/en

Building With Jenkins

- Spent years building with Jenkins on Proxmox/Hetzner
- Could never really get the configuration the way I wanted so PRs into the repository would trigger builds

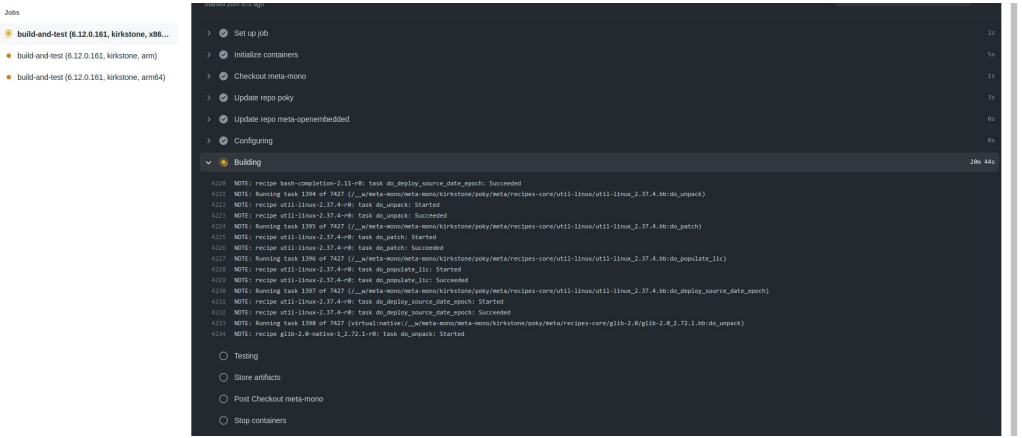


Using GitHub Actions

- Discovered GitHub Actions. They are really easy to use!

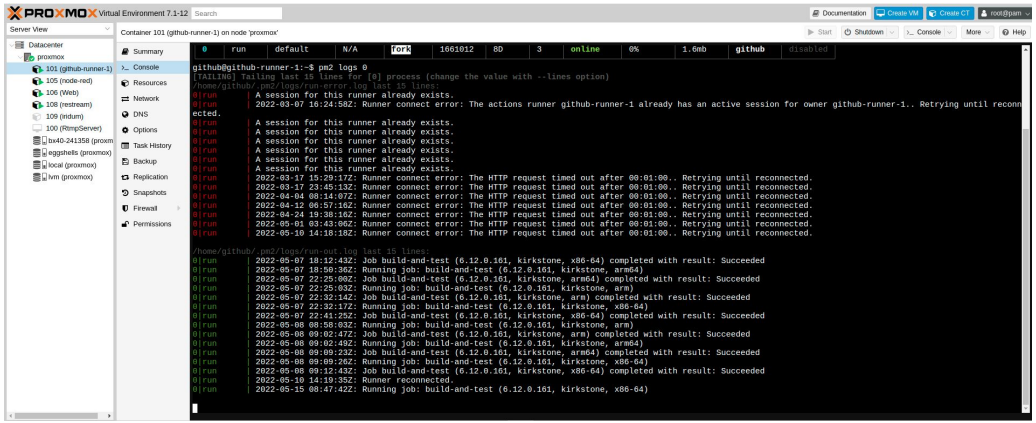
83 lines (78 sloc) | 3.26 KB

```yaml
name: meta-mono

on:
  push:
    branches:
      - master
      - master-next
  pull_request:

jobs:
  build-and-test:
    runs-on: [self-hosted, linux, X64]
    container:
      image: dynamicdevices/yocto-ci-build:latest
      options: --privileged --platform linux/amd64  -v /dev/net/tun:/dev/net/tun -v /dev/kvm:/dev/kvm
    strategy:
      matrix:
        mono_version: [6.12.0.161]
        branch: [kirkstone]
        arch: [x86-64, arm, arm64]
    env:
      name: build-and-test
      MONO_VERSION: ${{ matrix.mono_version }}
      ARCH: ${{ matrix.arch }}
      BRANCH: ${{ matrix.branch }}
```



GitHub Self-Hosted Runners

- The final piece of the puzzle was self-hosted runners
- This is a “stub” on my own hardware which connects into GitHub and runs my actions on certain triggers like PR



The CI / QEMU Testing Stack

`bitbake -c testimage` **runs tests inside QEMU**

`bitbake` **builds inside Docker targeting QEMU archs.**

Docker hosting a Yocto build image I created

Container for self-hosted-runner (6 cores, 6GB atm)

Proxmox Virtualisation Environment on Hetzner box

Hetzner physical hardware in the Cloud (8 cores, 32GB)

GitHub Self-Hosted Runners

Live… looking at how this all works

Reference Links

- GitHub Actions
  https://docs.github.com/en/actions

- GitHub Self-Hosted Runners
  https://docs.github.com/en/actions/hosting-your-own-runners/about-self-hosted-runners

Next Steps: CD and on-board testing

- The build artifacts can be uploaded and stored so currently I archive `tmp/deploy/$board/images`
- In the future I am considering archiving all packages
- I would also love to move the QEMU testing onto a real board rack for automated build and test IRL

THAT'S ALL FOLKS !!!

yocto PROJECT | THE LINUX FOUNDATION